بسم الله الرحمن الرحيم

الحمد لله رب العالمين، القائل في محكم التنزيل (وعلمك ما لم تكن تعلم ، وكان فضل الله عليك عظيماً) ، والصلاة و السلام على سيدنا محمد سيد العلماء و سيد الأولين و الآخرين رسول رب العالمين ، وعلى آله و صحبه أجمعين .

(سبحانك لا علم لنا إلا ما علمتنا إنك أنت العليم الحكيم)

هذا الكتاب موجه إلى كل من مبتدئ في البرمجة وإلى طلابنا في المعاهد و الجامعات لتكون عوناً لهم إن شاء الله .

وأدعو الله أن يتقبله خالصاً لوجه وأن يكتبه في صحيفتي و صحيفة من سيتابع في نشره وأن يجعله عملاً ينتفع به إلى يوم الدين

فمن وجد خطأً فهو مني وما كان فيه من صواب فمن توفيق الله

وآخر دعوانا الحمد لله رب العالمين

أخوكم في الله
محمد هنداوي

mh2n@scs-net.org

المحتويات

لغة C++

تسمية المتغيرات في لغة C++

١- يجب أن لا تكون أسماء المتغيرات من
الكلمات المحجوزة (RESTRAINED WORDS)
أو الكلمات التي تحمل معناً خاصاً مثل main
٢- يمكن أن يحتوي الاسم على أي حرف من الحروف الأبجدية صغيرة أو كبيرة وعلى أي رقم من الأرقام كذلك يمكن أن تحوي الإشارة (_)

٣- لا يجوز أن يبدأ الاسم برقم
مثال:

7-up	خطأ بسبب البداية برقم
Salim!	خطأ بسبب استخدام (!) المحجوزة
No#	خطأ بسبب استخدام (#) المحجوزة
A9	عبارة صحيحة
a-a	خطأ بسبب إشارة الطرح

٤-إن هذه اللغة حساسة لحالة الحرف أي الحرف الصغير ليس هو الحرف الكبير
فإن المتحول a فهو غير المتحول A

كيفية كتابة برنامج بلغة C++

لكتابة برنامج يجب أن نستهل البرنامج بالمكتبات التي تدعم التعليمات الواجب استخدامها و المكتبة التي سنستخدمها الآن هي مكتبة iostream
بحيث تمكننا هذه المكتبة من استخدام الأوامر التالية

cin>>	إدخال وتقرأ	c in
cout<<	إخراج و تقرأ	c out
endl	نهاية سطر وتقرأ	end line

تتعامل هذه لغة C++ مع عدد كبير من المكتبات وكل مكتبة لها اكواد خاصة بها و يمكن التعامل مع عدة مكتبات في آن واحد وسنتحدث عن كل مكتبة عندما يتطلب البرنامج ذلك

كيفية كتابة البرنامج

يكتب أي برنامج كما في الشكل التالي:

#include <iostream.h>	مقدمة لذكر اسم المكتبة
Void main( )	مقدمة أيضا سنشرح هذه الأوامر في الفصول اللاحقة
{	قوس الدلالة لبداية البرنامج
نبدأ بالإعلام عن المتغيرات	
يكتب هنا البرنامج	
}	قوس الدلالة على نهاية البرنامج

الإعلام عن المتغيرات

ليتمكن المستخدم من استخدام المتغيرات التي يريدها يتطلب البرنامج الإعلام المسبق عن أسماء هذه المتغيرات فمثلاً للتعامل مع متغيرين من نوع (عدد صحيح) نكتب ما يلي:

```
int a;
int b;
int a,b ;    ويمكن أن تكتب بالشكل التالي
```

وسنستخدم البرنامج Turbo لتنفيذ الأمثله كما هو ظاهر في الشكل :



ويكفيك أن تضغط القائمة File وتختار الأمر New وتبدأ بكتابة البرنامج وتضغط الأمر Run

لكي تنفذ البرنامج

مثال:

اكتب برنامج يقوم بالتصريح عن متحولين صحيحين ومن ثم يقوم بجمع قيمة الأول إلى عشرة وإسناده إلى قيمة المتحول الثاني ومن ثم يقوم البرنامج بطباعة قيمة الناتج ؟

```
#include <iostream.h>
void main ( )
{
int vor1;
int vor2;
cin>> var1;
var2 = var1 + 10;
cout<< " var1+ 10 is "<<var2;
}
```

العبارة *cin>> var1;* تعني أدخل قيمة المتحول *var1*

أنواع المتغيرات :

المتغيرات الحرفية char

وتتضمن الحروف بكافة أشكالها و الرموز و الفراغات (مسافة فارغة) مثل :

```
char a,b;
a= 'a' ;
b = ' ' ;
char var1;
var1 = '.' ;
```

المتغيرات الصحيحة integer

تتضمن قيم عددية صحيحة يمكن أن تأخذ قيمة تصل إلى ٣٢٧٦٧
وتكتب على الشكل التالي :

```
int a;
a= 100;
```

المتغيرات العددية العشرية floating point

تتضمن جميع الأعداد الحقيقية
وتكتب على الشكل التالي:

```
float x;
x = 5.2;
```

المتغيرات العددية العشرية الطويلة double

هي نفس المتغيرات العددية العشرية ولكن يمكن تمثيلها إلى خمسة عشر خانة
وتكتب على الشكل التالي:

```
double x;
```

ملاحظات هامة
قســـمة عدد صحيح على عدد صحيح يكون النـــاتج صحيح أيضاً
قســـمة عـدد حقيقي على عدد صحيح يكون الـــنـــاتج حقيقي أيضاً
أما قسمة عدد حقيقي على عدد صحيح يكون الـنـاتـج حقيقي

لا توجد عملية قسمة عدد صحيح على عدد حقيقي

مثال :
أوجد نتيجة التمرين التالي

```
#include <iostream.h>
void main ( )
{
int  i , j ;
```

```
float x,y,z;
i = 5/2;
x = 5/2 ;
y = (float ) (5/2);
j = ( float) 5/2;
z = 5. / 2 ;
cout<<i <<endl<< x<<endl << y<<endl << j<<endl << z<<endl;
}
```

الجواب

I=2 x= 2.000 y= 2.000 j=2.5

العمليات الحسابية و الأدوات المستخدمة فيها

للجمع تستخدم +
للطرح -
للقسمة /
باقي القسمة %
للضرب *
للزيادة بمقدار واحد ++
للنقصان بمقدار واحد --

أولويات العمليات الحسابية

١- الزيادة و النقصان عندما تأتي قبل العدد
٢- الأقواس
٣- إشارة السالب
٤- القسمة و باقي القسمة و الضرب
٥- الجمع و الصرح
٦- المساواة
٧- الزيادة و النقصان المتأخرة بعد العدد

ملاحظة

في حال وجود عمليتين لهما نفس الأولوية نبدأ بتنفيذ العملية الأقرب إلى اليسار.

الأدوات العلاقية :

وهي حسب هذا الجدول

الأكبر	<
الأصغر	>
أكبر أو يساوي	<=
أصغر أو يساوي	>=
إن كان يساوي	==
إن كان لا يساوي	!=

أما بالنسبة للأولويات فهي على نفس الترتيب

مثال:

أكتب برنامج لإدخال عدد ما ومن ثم طباعة جدول ضرب له

الحل:

```
#include < iostream.h>
void main ( )
}
int x;
cin>> x;
cout<< x <<endl;
cout<< x *2<<endl;
cout<< x *3<<endl;
cout<< x *4<<endl;
cout<< x *5<<endl;
cout<< x *6<<endl;
cout<< x *7<<endl;
cout<< x *8<<endl;
cout<< x *9<<endl;
{
```

مثال آخر:

اكتب برنامج لإدخال ثلاثة علامات لطالب ومن ثم طباعة معدل هذه العلامات

الحل:

```
#include <iostream.h>
void main ()
{
int a,b,c;
cin>> a >> b >> c ;
cout<<"the averaje is"<< ( a+b+c )/3;
}
```

الأدوات المنطقية

&&	and
||	or
!	not

مثال:

```
Int a=b=3;
A < 3      false
A < = 3    true
A > b      false
A != b     false
A = =b     true
```

الأدوات الدقيقة :

تتميز لغو C++ أنها تستخدم أدوات دقيقة تتعامل مع bit وتستعمل هذه الأدوات مع المعطيات char , Int ولا تستخدم مع غيرها.

وهذه الأدوات هي:

التعليمة	الشرح
~	Not
>>	إزاحة الى اليسار
<<	إزاحة الى اليمين
^	Xor
&	And
|	Or

وتكون الأولوية حسب ترتيب الجدول.

مثال:

إذا كان لدينا x ممثل بالنظام الثنائي

0	0	0	1	1	0	1	0

المطلوب أوجد قيمة نفي x أي ~x

الجواب:

1	1	1	0	0	1	0	1

الأدوات الشرطية:

وهي عبارة عن حلقات تتم ضمن شروط يحددها كاتب البرنامج

الأدوات الشرطية الأولية conditional operator

الشكل النموذجي هو :

Expression ? Expression1 : Expression2
(a>b)? c : d

ويكون الجواب إن كان الشرط صحيح فاختر القيمة c
وإن كان الشرط غير صحيح فاختر d

مثال :

اكتب برنامج لحساب القيمة المطلقة للمتحول y باستخدام المعامل الشرطي (الأداة الشرطية)
و المعرفة بالعلاقة التالية

y تساوي x بالقيمة المطلقة

أي أن y = x عندما x>0

و y= -x عندما x > 0

الحل:

```
#include < iostream.h>
void main ( )
}
```

```
int y, x;
cin>> x;
y =(x >= 0)? x :-x;
cout<< y;
{
```

الجملة الشرطية if – else

شكلها العام :

```
if(Expression)
Expression1
else
Expression2
```

مثال توضيحي

```
if(b < c)
a = b;
else
a = c;
```

مثال آخر

اكتب برنامج لإدخال طولك و طول زميلك بشرط
إن كان طولك أكبر من طول زميلك عندئذ قم بطباعة طولك ومن ثم قم بحساب معدل الأطوال
ثم أطبعه وإلا فأطبع طول زميلك وأطبع ضعف طوله وأطبع نصف طوله

الحل:

```
#include <iostream.h>
void main()
{
int z,y;
cout<< "Inter your long"<<endl;
cin>> z ;
cout<< "Inter friend's long"<<endl;
cin>> y;
cout<< "--------------------"<<endl;
if(z>y)
{
cout<< "You longer"<<endl;
cout<< "Your long is   "<<z<<endl;
cout<<endl;
cout<< " The average longs are   ";
```

```
cout<< (z+y)/2;
}
else
{
cout<<" Your friend is longer"<<endl;
cout<<"   He's long is    "<<y<<endl;
cout<<" y*2 = "<<y*2<<endl;
cout<<" y/2 = "<<y/2;
}
}
```

مثال آخر:

ناقش حالة إدخال عددين بمختلف الحالات

```
#include <iostream.h>
void main()
{
int z,y;
cout<< "enter first number"<<endl;
cin>> z;
cout<<"enter scanned number"<<endl;
cin>>y;
if(z>y)
cout<<"the large number is first"<<z<<endl;
else
if(z==y)
cout<<"z=y";
else
cout<<"the large number is  scaned";
}
```

حلقة (switch – case)

شكلها العام

```
switch(Expression)
```

```
{
case constant 1 : statement 1;
                  break;
case constant 2 : statement 2;
                  break;
                  .
                  .
                  .
case constant x : statement x;
                  break;
default :          statement;
                  break;
}
```

ملاحظة :

يجب أن يكون Expression المعرف في switch قيمة صحيحة

مثال:

```
#include <iostream.h>
void main()
{
int x;
cin>> x;
switch (x)
{
case 1:
cout<<" welcome"<<endl;
break;
case 2:
cout<<" hello student"<<endl;
break;
case 3:
cout<<" introdaction to c++"<<endl;
break;
default:
cout<< "bye bye";
}
}
```

مثال آخر:

اكتب برنامج يطبع اسم الكوكب في المجموعة الشمسية مرتباً من الأقرب إلى الشمس بحيث ندخل إلى البرنامج رقم الكوكب فيطبع البرنامج اسم الكوكب

الحل

```
#include <iostream.h>
void main()
{
int x;
cout<< " inter number a planet "<< endl;
cin>> x;
switch (x)
{
case 1:
		cout<<" its a  Mercury"<<endl;
		break;
case 2:
		cout<<" its a  Venus"<<endl;
		break;
case 3:
		cout<<" its a  earth"<<endl;
		break;
case 4:
		cout<<" its a  Mars"<<endl;
		break;
case 5:
		cout<<" its a  Jupiter"<<endl;
		break;
case 6:
		cout<<" its a  Saturn"<<endl;
		break;
case 7:
		cout<<" its a  URANUS"<<endl;
		break;
case 8:
		cout<<" its a  NEPTUNE"<<endl;
		break;
case 9:
		cout<<" its a  PLUTO"<<endl;
		break;
case 10:
		cout<<" its a  senda "<<endl;
		break;
```

```
default:
                cout<< "err no plant to resemble this number"<<endl;
                break;
                }
                }
```

مثال آخر:
اكتب برنامج يحول من الأرقام العربية الأصل إلى الأرقام الموافقة لها بالرومانية
الحل:

```
#include<iostream.h>
main()
{
int x;
cout<<"enter the decimal number:";
cin>> x;
while(x<=12)
{switch(x)
{
case 1:
{cout<<"I"<<endl;
cout<<"enter the decimal number:";}
break;
case 2:
{cout<<"II"<<endl;
cout<<"enter the decimal number:";}
break;
case 3:
{cout<<"III"<<endl;
cout<<"enter the decimal number:";}
break;
case 4:
{cout<<"IV"<<endl;
cout <<"enter the decimal number:";}
break;
case 5:
{cout<<"V"<<endl;
cout<<"enter the decimal number:";}
break;
case 6:
{cout<<"VI"<<endl;
 cout<<"enter the decimal number:";}
```

```
break;
case 7:
{cout<<"VII"<<endl;
 cout<<"enter the decimal number:";}
break;
case 8:
{cout<<"VIII"<<endl;
 cout<<"enter the decimal number";}
break;
case 9:
{cout<<"IX"<<endl;
 cout<<"enter the decimal number";}
break;
case 10:
{cout<<"XI"<<endl;
 cout<<"enter the decimal number";}
break;
case 11:
{cout<<"XII"<<endl;
 cout<<"enter the decimal number";}
break;
case 12:
{cout<<"XIII"<<endl;
 cout<<"enter the decimal number:";}
break;
default:
cout<<"NOOOO";
break;
}
cin>>x;
}
}
```

حلقة for

شكلها العام

```
For ( initial value ; coalition ; increment )
        Statements;
```

الشرح : عبارة عن شرط يقيد حركة for وغالباً ما يحوي قيمة نهائية
و هذا الشرط يكتب فيه القيمة الابتدائية و القيمة النهائية و قيمة الزيادة للمتحول .

مثال:

اكتب برنامج يطبع هذه الإشارة * عشر مرات متتالية

```
 #include<iostream.h>
main()
{
int i;
for ( i = 1 ; i < = 10 ; i + + )
cout<< "*";
}
```

مثال آخر:

اكتب برنامج يقوم بطباعة الأعداد الفردية من الرقم (١) إلى الرقم (١٥) .

الحل:

```
#include <iostream.h>
 void main()
 {
 int x;
 for(x=1; x<=15 ; x=x+2)
 cout<<x<<endl;
 }
```

ملاحظة :

عند مقدار الزيادة واحد نكتب (i++) أما عند الزيادة اثنان اثنان نكتب (I=I+2)
وبإمكاننا أن نعرف المتحول داخل الحلقة أيضاً مثل for(int i=1; i=15 ; i=1+2)

مثال آخر:

اكتب برنامج يطبع الارقام 10.......2.3 بشكل عامودي

الحل :

```
#include <iostream.h>
 void main()
 {

 for(int x=2; x<=10 ; x++)
 cout<<x<<endl;
 }
```

حلقات for المتداخلة

يمكن أن تأتي حلقة for تابعة لحلقة for أخرى أي متضمنة بداخلها على الشكل التالي

```
For ( I =1; I < = 10 ; I ++)
For ( j = 2 ; j < = 20 ; j ++)
```

و هنا تنفذ العملية الثانية حسب الحلقة الأولى
أي كل مرة تنفذ فيها الحلقة الأولى تنفذ الحلقة الثانية من البداية إلى النهاية

مثال :

اكتب نتيجة البرنامج التالي

```
#include <iostream.h>
 void main()
 {
  int j ,i;
 for(i=1; i<=3 ; i++)
 for(j=1; j<=4 ; j++)
```

```
{
cout<<i;
cout<<j;
}
}
```

الجواب :

11121314   21222324   31323334

حلقة while

شكلها العام

```
while(condition)
Statement ;
```

ملاحظة :

يجب إضافة تعليمة في نهاية هذه الحلقة لتسمح العودة إلى بداية البرنامج حتى يتم فحص الشرط مرة أخرى

مثال:

اكتب برنامج يقوم المستخدم بإدخال عدد ما ومن ثم يفحص البرنامج هذا العدد لمعرفة فيما إذا كان هذا العدد أولي أم لا

الحل:

```
#include <iostream.h>
void main()
{
int x,i;
cin>> x;
i=2;
while (x>i)
{
if (x%i==0)
{
cout<<"the number in not  prime";// make i  right any thing
```

```
i=x+1;
}
else
if(x % i !=0)
i++;
}
if(x==i)
cout<<"the  number is  prime";
}
```

مثال آخر:

اكتب برنامج يطلب من المستخدم إدخال قيمة عددية ما

وطالما كانت هذه القيمة موجبة فيطبع البرنامج هذه الإشارة * على سطر جديد .

الحل

```
#include <iostream.h>
void main()
{
int x,i;
cin>> x;
while (x>0)
{
cout<<"*"<<endl;
cin>>x;
}
}
```

وهنا يبقى البرنامج في الحلقة ولا يخرج منها إلا بكتابة قيمة سالبة

أما إن اردنا أن يخرج البرنامج بعد أول مرة فنكتب هذه العبارة

x=x-x;

بدلاً من العبارة

Cin>>x;

حلقة do while

الشكل العام

```
Do
{
Statement
}while (condition);
```

الشرح:

ابق ضمن الحلقة do حتى يتم تنفيذ الشرط while (condition)

<u>ملاحظة هامة جداً</u>

الفرق الجوهري بين الحلقة do while و الحلقة while

<u>هو أن البرنامج في الحلقة do while ينفذ مرة واحدة على الأقل</u>

مثال

اكتب برنامج يقوم بجمع الأعداد المدخلة ومن ثم يتوقف عن العمل ويقوم بطباعتها بمجرد إدخال الرقم صفر.

```
#include <iostream.h>
void main()
{
int x, sum;
sum = 0;
do
{
cin>>x;
sum+=x;
}while (x!=0);
cout<<"sum of number is ="<<sum;
}
```

ملاحظة :

إن العبارة sum+=x تكافئ و تساوي العبارة sum = sum + x

ملاحظة أخرى

يمكن كتابة هذا البرنامج بالحلقة while فقط على النحو التالي

```
#include <iostream.h>
void main()
{
int x, sum;
sum = 0;
do
{
cin>>x;
While ( x!=0)
{
Sum+=x;
Cin>>x;
}
```

حلقة الإيقاف break

وظيفتها : إيقاف بنية أو حلقة تكرار عند تحقق شرط أو شروط معينة .
عند تنفيذ هذه التعليمة يتم القفز إلى سلسلة الجمل التالية للبنية أو حلقة التكرار.
مثال:

```
 #include <iostream.h>
void main()
{
for(int i=1 ; i<= 100 ; i++ )
{
cout<<i;
if(I = = 10)
break;
}
}
```

حلقة الاستمرار continue

تعمل هذه الحلقة على تجاوز تنفيذ بقية التعليمات في التكرار خلال الدورة الحالية و الانتقال إلى الدورة التالية .
مثال:

```
#include <iostream.h>
void main()
{
```

```
for(int i=1; i<=100; i++)
{
if(i==10)
continue;
cout<< i <<endl;
}
}
```

مثال آخر:

اكتب برنامج يقوم بطباعة جميع الأرقام الواقعة بين (١) إلى (١٠٠) التي تقبل القسمة على الأعداد التالية (٢و٤و٦) .

```
#include <iostream.h>
void main()
{
for(int i=1; i<=100; i++)
{
if(i%6=0 && i%4=0 && i%2=0)
cout<< i<<endl;}}
```

طريقة أخرى أسهل:

```
#include <iostream.h>
void main()
{
for(int i=1; i<=100; i++)
{
            if(i%2!=0 )
            continue;
else   if(i%4!=0 )
            continue;
else   if(i%6!=0 )
            continue;
cout<<i<<endl;
}
}
```

تعليمة الانتقال goto

تفيد هذه العملية في الانتقال من سطر برمجي إلى آخر مع تجاوز لبعض الأسطر البرمجية من أجل تنفيذ أمر معين.

مثال:

اكتب برنامج يقوم بإدخال رقم ما فإن كان الرقم أصغر من عشرة فيطبع البرنامج العبارة التالية

The number is less ten

أما إن كان الرقم أكبر أو يساوي العشرة فيطبع البرنامج العبارة

The number is equal or beggar ten

```
#include <iostream.h>
void main()
{
int x;
cin>>x;
if(x<10)
goto one;
else goto two;
one: cout<<"the number is less ten";
two: cout<<"the number is equal or beggar ten";}
```

ولكن نجد أن البرنامج يعطي العبارتين معاً إن كان الرقم المدخل أصغر من عشرة
و هذا الخطأ سببه:
البرنامج يقوم بتنفيذ التعليمات سطر سطر وبما أن العبارة الثانية جاءت تماماً بعد العبارة الأولى (في السطر التالي لها) فنفذت بطبيعة الحال.

تدريبات:

اكتب برنامج يقوم المستخدم به بإدخال عدة أعداد / n / ويقوم بجمعها بحيث يحدد المستخدم عدد الأعداد /n/ قبل إدخالها ؟
اختر الطريقة المناسبة لحل هذا المثال؟

الحل الطريقة الأولى باستخدام الحلقة do while

```
#include <iostream.h>
void main()
{
int n,x,sum;
sum=0;
cin>>n;
do
{
cin>>x;
sum+=x;
n--;
}
while(n!=0);
cout<<"the sum of number is "<<sum;
```

```
}
```

الطريقة الثانية باستخدام الحلقة while

```
 #include <iostream.h>
void main()
{
int n,count=1,num,sum=0;
cout<<"enter n";
cin>>n;
while(count<=n)
{
cin>>num;
sum+=num;
count++;
}
cout <<"the sum of number is"<<sum;
}
```

الطريقة الثالثة باستخدام الحلقة for

```
#include <iostream.h>
 void main()
 {
 int sum=0,x,i,n;
 cout<<"enter n"<<endl;
 cin>>n;
 for(i=1; i<=n ; i++)
                  {
                  cout<<"enter namber"<<endl;
                  cin>>x;
                  sum=sum+x;
                  }
cout<<" the sum of number is = ";
cout<<sum<<endl;
}
```

اكتب برنامج يمثل آلة حاسبة بسيطة تدخل عددين فقط بحيث نختار عملية من العمليات الحسابية الأربعة مثل أن نكتب ١+٢ فيعطينا البرنامج الجواب
وناقش حالة القسمة على صفر؟

```
#include<iostream.h>
void main()
{
int x,y;
```

```
char r;
cin>> x;
cin>> r;
cin>> y;
switch(r)
{
case '+':
                    cout<<x+y;
                     break;
case '-':
                    cout<<x-y;
                    break;
case '*':
                    cout<<x*y;
                    break;
case '/':
                    if(y==0)
                    cout<<"error";
                    else
                    cout<<x/y;
                    break;
default: cout<<" chose the right operate";}}
```

مثال مماثل ولكن بتطوير أكثر

اكتب نفس البرنامج السابق ولكن اسمح للبرنامج بأن يدخل العمليات أكثر من مرة

```
#include<iostream.h>
void main()
{
int x,y,n,count;
char r;

             cout<<" enter how many play program"<<endl;
             cin>> n;
             cin>> x;
             cin>> r;
             cin>> y;

 while(n!=0)
 {
 n--;
       switch(r)
             {
             case '+':
```

```
                              {
                              cout<<"the score is = "<<x+y<<endl;
                              cout<<"----------------------"<<endl;
                               }
                               break;
           case '-':
                              {
                              cout<<"the score is = "<<x-y<<endl;
                              cout<<"----------------------"<<endl;
                              }
                              break;
           case '*':
                              {
                              cout<<"the score is = "<<x*y<<endl;
                              cout<<"----------------------"<<endl;
                              }
                              break;
           case '/':
                              {
                              cout<<"the score is = ";
                              if(y==0)
                              cout<<"error"<<endl;
                              else
                              cout<<x/y<<endl;
                              cout<<"----------------------"<<endl;
                              }
                              break;
           default:
                              {
                              cout<<" chose the right operate"<<endl;
                              cout<<"----------------------"<<endl;
         }
           }
           cin>> x;
           cin>> r;
           cin>> y;
}
}
```

مثال:

اكتب برنامج يقوم بطباعة الشكل التالي

1
1 2
1 2 3

1 2 3 4
1 2 3 4 5

الحل:

باستخدام علاقة رياضية أي يكون الخرج رقماً وليس شكل هو كالآتي:

```
#include<iostream.h>
void main()
{
int x;
x=0;
for(int i=1;i<=5 ;i++)

{
x=x*10+i;
cout<<x<<endl;
}
}
```

أما كتابة البرنامج بطريقة يعطينا الخرج شكلاً وليس رقم فهو:

```
#include<iostream.h>
void main()
{
char c= ' ';
for(int i=1;i<=5;i++)
        {
        for(int j=1;j<=i;j++)
                cout<<j<<c;
     cout<<endl;
        }
}
```

ملاحظة : يمكن أن نكتب في السطر الواحد أكثر من أمر برمجي ولكننا فضلنا أن لا نفعل ذلك من أجل الترتيب و فهم البرنامج مع أننا قمنا بذلك في بعض الأمثلة

فيجوز مثلاً كتابة cin>>x;cin>>y; على سطر واحد وهكذا .

مثال:

اكتب برنامج يقوم بحساب n! العاملي لأي رقم

الحل :

باستخدام الحلقة while

```
#include<iostream.h>
void main()
{
int x,y,fact=1;
cout<<" this program is for factorial calculation";

cout<<"\n ----------------------------------------";
cout<<"\n enter x"<<endl ;
cin>>x;
y=1;
cout<<x<<"! =  ";
while(y<=x)
          {
                  fact=fact*y;
                  y++;
                  }
                  cout<<fact;
}
```

ويمكن كتابة البرنامج باستخدام الحلقة for على الشكل التالي

```
#include<iostream.h>
void main()
{
int i,n,fact=1;
cout<<" this program is for factorial calculation";
cout<<"\n copy right 2006";
cout<<"\n designed by hindawi";
cout<<"\n ----------------------------------------";
cout<<"\n enter x"<<endl ;
cin>>n;
cout<<n<<"! =  ";
for(i=1;i<=n;i++)
        {
        fact=fact*i;
        }
        cout<<fact;
}
```

Fibonacci Series سلسلة فيبوناشي

هي عبارة عن سلسلة أعداد يون أي رقم فيها يساوي مجموع الرقمين السابقين له و يكون الرقم الأول منها و الثاني يساوي الواحد .

أي كما في هذه السلسلة

1 1 2 3 5 8 13 21 ……………..n

مثال

اكتب برنامج يقوم بحساب العناصر العشرة الأولى من سلسلة فيبوناشي

```
#include<iostream.h>
void main()
{
int a,b,c,i;
char s=' ';
a=b=1;
cout<<a<<s<<b<<s;
for(i=3;i<=10;i++)
{
c=a+b;
cout<<c<<s;
a=b;
b=c;
}
```

```
}
```

مثال آخر:

أوجد حدود فيبوناشي من أجل n

الحل:

```
#include<iostream.h>
void main()
{
int a,b,c,i,n;
char s=' ';
a=b=1;
cin>>n;
cout<<a<<s<<b<<s;
for(i=3;i<=n;i++)
{
c=a+b;
cout<<c<<s;
a=b;
b=c;
}}
```

تدريبات

اكتب برنامج يقوم بطباعة الشكل التالي

1 2 3 4 5
2 3 4 5 1
3 4 5 1 2
4 5 1 2 3
5 1 2 3 4

الحل:

```
#include<iostream.h>
void main()
{
int a,b,c,d,e;
char s= ' ';
a=1;
b=2;
c=3;
d=4;
e=5;
for(int i=1;i<=5;i++)
    {
    cout<<a<<s<<b<<s<<c<<s<<d<<s<<e<<endl;
```

```
    a=b;
    b=c;
    c=d;
    d=e;
    e=a;
    e--;
    }
}
```

مثال:

اكتب برنامج يقوم بطباعة الشكل التالي على الشاشة

A.1 A.2 A.3 A.4 A.5
B.1 B.2 B.3 B.4 B.5
C.1 C.2 C.3 C.4 C.5
D.1 D.2 D.3 D.4 D.5
E.1 E.2 E.3 E.4 E.5

الحل

```
#include<iostream.h>
void main()
{
char A,B,C,D,E,y,s,n;
s='.';
n=' ';
for(y='A';y<='E';y++)
    {
    cout<<endl;
    for(int i=1;i<=5;i++)
    cout<<y<<s<<i<<n;
    }}
```

مثال:

اكتب برنامج يقوم بحساب N لهذه المتوالية

1 >2>4>8>16>32> >>>>>>>>>>

الحل

```
#include<iostream.h>
void main()
{
float a,b,n;
char s=' ';
a=b=1;
cin>>n;
```

```
cout<<a<<s<<b<<s;
for(int i=3;i<=n;i++)
{
b=2*a;
cout<<b<<s;
a=b;
}
}
```

المصفوفات Arrays

المصفوفة هي عبارة عن مجموعة من البيانات التي تتشابه في النوع double , int , char , float ولها اسم مشترك .

- تتشابه في النوع : يعني أن تكون البيانات التي تخزنها المصفوفة مثلاً كلها أعداد صحيحة.
- لها اسم مشترك : يعبر عن هذه البيانات باسم المصفوفة

التصريح عن المصفوفة:

هناك ثلاثة أشياء يجب أن تأخذ بعين الاعتبار عند التصريح عن المصفوفة

- اسم المصفوفة : وهو اسم نختاره مثلما نختار اسم أي متغير.
- عدد العناصر التي بداخلها.
- نوع البيانات المستخدمة فيها.

ويمكن أن تكتب كما في هذه الأمثلة:

```
Int mark [5] ={1,3,2,4,6};
Int mark [ ] = {5,2,1};
```

```
Int mark [5];
```

حيث:

int نوع البيانات التي بداخل المصفوفة
mark اسم المصفوفة
[5] عدد العناصر داخل المصفوفة

علماً أنه في المصفوفات يبدأ ترقيم العناصر (خانات المصفوفة) وليس قيمتها بالرقم صفر
أي وبشكل أوضح:
مصفوفة تتألف من ستة عناصر فإنها تتمثل بالشكل

العنصر ٥	العنصر ٤	العنصر ٣	العنصر ٢	العنصر ١	العنصر ٠
قيمة	قيمة	قيمة	قيمة	قيمة	قيمة

ومن هذا الشكل نستنتج بأن المصفوفة يبدأ ترقيمها من الصفر

كيفية إدخال عناصر مصفوفة

يتم ذكر اسم المصفوفة ثم رقم العنصر الذي نريد التعامل معه بين قوسين .
فإذا أردنا مثلاً أن نضع القيمة ٧٥ في العنصر الثالث من المصفوفة يتم ذلك كما يلي

Mark[2] = 75

ملاحظة هامة جداً
البداية في المصفوفة تبدأ من الصفر (أي المؤشر) كما ذكرنا سابقاً.
كما في الشكل التالي

4	3	2	1	0	رقم العنصر
852	258	1	11	454	قيمة العنصر

فهذه المصفوفة تتألف من خمسة عناصر.
لنتمكن من إدخال قيمة عنصر-حرف أو رقم - في مصفوفة يجب علينا تحديد رقم العنصر حتى نتمكن من ذلك كما في المثال التالي:
أدخل القيمة ٣ في المكان الثاني للمصفوفة mark[4]

الحل:

cin>>mark[1];

طبعا كتبنا [1] لأن العنصر الثاني من المصفوفة يشار له بالرقم ١
ولأن العنصر الأول منها يأخذ المكان (0) كما تم ذكره سابقاً

كيفية إدخال عناصر مصفوفة بشكل متتالي:

لإدخال عناصر مصفوفة اسمها mark تتألف من خمس عناصر نتبع حلقة for كما يلي:

```
for(int i=0 ; i<5 ; i++)
{
cout<<" please enter array elements" ;
cin>> mark [i] ;
}
```

* علماً أن رقم العنصر (وليس قيمته) يبدأ من الرقم صفر ولهذا كتبنا i<5
فالمصفوفة التي تتألف من خمسة عناصر يكون رقم العنصر الأخير – وليس قيمته – تكون أربعة

تنقسم المصفوفة إلى نوعين :

١- مصفوفة ذات البعد الواحد
٢- مصفوفة ذات بعدين

وتسمى المصفوفة A{ 2,5,4,6} مصفوفة ذات البعد الواحد لأنها تتكون من صف واحد و إن كانت تتكون من عامود واحد فتسمى نفس الاسم .
أما المصفوفة ذات البعدين فهي التي تتألف من أكثر من صف أو أكثر من عامود.

ملاحظة :
قد نحتاج في المصفوفات أو غيرها للتعامل مع قيمة ثابتة (مثلاً ١٠) لمتحول ما(max) لا تتغير أبدا أي متحول ثابت القيمة فإننا نعرف هذا المتحول على الشكل التالي

```
#define max 10
```

ففي هذه التعليمة فإن المتحول max أخذ القيمة (١٠) طوال فترة تنفيذ البرنامج
ولا يمكن أن يدخل max في أي من العمليات الحسابية أو الإسناد.

مثال :اكتب برنامج يقوم بإدخال مرتبات عشرة موظفين ثم يقوم بحساب وطباعة متوسط هذه الرواتب

```
#include<iostream.h>
#define max 10          تعريف المتحول الثابت
void main()
```

```
{
float salary[max];
float average ,sum;
int count;
sum=0.0;
for(count=0;count<max;count++)        كيفية استخدام المتحول الثابت ضمن هذه الحلقة
    {
    cout<<"please enter salary for employee\n";
    cin>>salary[count];
    sum=sum+salary[count];
    }
average = sum / max;        استخدام المتحول الثابت ضمن العمليات الحسابية
cout<<"\n salary average is = "<<average;
}
```

ملاحظة

* تعرفنا سابقا على نوع البيانات char حيث يمكن أن يضم حرف واحد فقط لكن في حال أردنا إدخال سلسلة من الأحرف على سبيل المثال :
نريد إدخال اسم موظف أو عنوان عندها سنقوم بالتعامل مع سلسلة من الأحرف أي : سيتم إدخال مجموعة من الأحرف.
وهذا ما يشبه مصفوفة من الأحرف و يسمى هذا النوع من أسلوب التعامل مع البيانات بـ string

* يمكن إدخال جميع عناصر المصفوفة الاسمية char دفعة واحدة كالتالي:

cin >> mark [ ] ;

ونكتب الاسم المراد ادخاله بالكامل ثم نضغط إنتر

بينما ليس بالإمكان فعل ذلك مع المصفوفة الرقمية float int إلا رقم رقم .

مثال

اكتب برنامج يقوم بإدخال أسماء الموظفين و عناوينهم

```
#include <iostream.h>
void main()
{
char name[20];
char address[20];
for(int i=0;i<13;i++)
    {
     cin>>name;
     cin>>address;
     cout<<name;
     cout<<address;
     }
 }
```

كما نلاحظ أننا في هذا المثال أدخلنا اسم الموظف كاملاً و عنوانه أيضاً لأن اسم الموظف هو عبارة عن مصفوفة حرفية وكذلك العنوان

اكتب برنامج من أجل إيجاد مجموع عناصر مصفوفة أحادية البعد عدد عناصرها خمسة عناصر

```
#include<iostream.h>
void main()
{
float mark[5];
int sum=0;
for(int i=0;i<5;i++)
    {
    cin>>mark[i];
    sum=sum+mark[i];
    }
cout<<"sum = "<<sum;
}
```

شرح البرنامج

لقد أدخلنا هنا العناصر رقم رقم و الذي قام بذلك حلقة FOR

ومن ثم قمنا بجمع قيم عناصر هذه المصفوفة

ملاحظة

ليس بالإمكان تحديد عدد عناصر المصفوفة بمتحول

فإنه من الواجب تحديد عدد عناصرها ولذلك فإنه من الخطأ كتابة

```
float mark[x]
cin>>x
```

لأنه لا يجوز تحديد عدد عناصر المصفوفة بمتحول

ملاحظة

لقد كتبنا في المثال السابق for(int i=0;i<5;i++) العبارة i<5 ولم نكتب i<=5 لأن عناصر المصفوفة تبدأ من الصفر

مثال آخر

اكتب برنامج يقوم بإدخال سلسلة مؤلفة من عشرة رموز ثم يقوم بطباعتها وفق ترتيب الإدخال ثم بعكسه

```
#include<iostream.h>
void main()
{
char s[10];                    حددنا عناصر المصفوفة بأنها عشرة عناصر
for (int i=0;i<10;i++)         حلقة لإدخال العناصر
      cin>>s[i];
```

```
for(i=0;i<10;i++)            حلقة الإخراج التصاعدية
     cout<<s[i];
for(i=9;i>=0;i--)            حلقة الإخراج التنازلية
     cout<<s[i];
}
```

ملاحظة هامة

* في هذا المثال أدخلنا عناصر المصفوفة حرف حرف

* لقد كتبنا حلقة الإخراج التنازلية بهذا الشكل

for(i=9;i>=0;i--)

للأسباب التالية   عندما i=9 تنعي بأننا نريد العنصر العاشر

عندما i=0 نعني بأننا نريد العنصر الأول

تمارين حول المصفوفة أحادية البعد

* اكتب برنامج يُمَكّن المستخدم من إدخال عشرة رموز ثم يقوم البرنامج بطباعة عدد مرات تكرار الحرفين a A في هذه السلسلة

```
#include<iostream.h>
#define n 10
void main()
{
int count=0;
char s[n];
for(int i=0;i<n;i++)
    {
    cin>> s[i];
    if(s[i]=='A'||s[i]=='a')
    count ++;

    }
cout<<count;
}
```

شرح البرنامج

كما تلاحظ

بعد أمر الادخال كتبنا الكود التالي

if(s[i]=='A'||s[i]=='a')

ما معنى هذه الحلقة ؟

إن كان العنصر المدخل يساوي القيمة A أو or يساوي القيمة a نفذ السطر التالي

والذي هو count++ أي زيد العداد بمقدار واحد
فكلما تنفذ هذا الشرط زادت قيمة العداد حتى تمام إدخال عناصر المصفوفة فإن البرنامج يطبع قيمة العداد.
مثال
اكتب برنامج يُمَكن المستخدم بإدخال مصفوفة أحادية البعد مؤلفة من عشرة أرقام
ومن ثم يقوم البرنامج بطباعة الأرقام التي هي أكبر من خمسة
مع طباعة أماكن تواجدها ضمن المصفوفة

الحل:

```
#include<iostream.h>
#define n 10
void main()
{
int i,s[n];
for(i=0;i<n;i++)
    {
    cin>>s[i];
    if(s[i]>5)
      {
      cout<<"this number is begger of 5"<<endl;
      cout<<i<<"          "<<s[i]<<endl;
      }
    }}
```

شرح البرنامج

إن القيمة I تمثل مكان تواجد العنصر داخل المصفوفة
أما القيمة S[I] فإنها تشكل قيمة هذا العنصر

المصفوفات ثنائية البعد

هي المصفوفات التي تتألف من أكثر من عامود أو أكثر من سطر
مثال عليها:

6	5
1	4
3	7

ففي هذا المثال تتألف هذه المصفوفة من عامودين و ثلاثة أسطر
ولإدخال هذه المصفوفة يجب علينا بداية أن نعرف البرنامج بسطر ثم بعامود العنصر المراد إدخاله
وتكون بالطريقة التالية:

```
cin>> a[i] [j];
```

حيث : a اسم المصفوفة

i رقم السطر
j رقم العامود
ويكون ترقيم الأسطر و الأعمدة حسب الشكل:

	عامود 0	عامود 1
سطر 0	6	5
سطر 1	1	4
سطر 2	3	7

حيث يعبر الكود a[1][0]=1 أن القيمة (١) موجودة في السطر 1 و العامود 0 كما في الشكل السابق
وكما أننا قد استخدمنا حلقة for في المصفوفة أحادية البعد فإننا سنستخدم هنا حلقتي for لإدخال المصفوفة ثنائية البعد كالتالي:

```
for(i=0 ; i<3 ; i++)
for(j=0 ; j<2; j++)
cin>>a[i][j];
```

ملاحظة:
نسمي المصفوفة ثنائية البعد بالمصفوفة المربعة إذا تساوى عدد الأسطر مع عدد الأعمدة

مثال :
اكتب برنامج يقوم بإدخال عناصر مصفوفة ثنائية البعد
تتألف من سطرين وثلاثة أعمدة
بحيث يقوم البرنامج باستبدال عناصر المصفوفة ذات القيمة الفردية بالقيمة (١٠)
ثم يقوم البرنامج بطباعة شكل المصفوفة الجديد.

الحل:

```
#include <iostream.h>
void main()
{
```

```
int s[2][3],i,j;
for(i=0;i<2;i++)
for(j=0;j<3;j++)
        {
        cin>>s[i][j];
        if(s[i][j]%2!=0)
                s[i][j]=10;
        }
for(i=0;i<2;i++)
        {
        for(j=0;j<3;j++)
        cout<<s[i][j]<<"  ";
        cout<<endl;

}}
```

السجلات (التركيبات) structure

تعريف السجل:
هو مجموعة من البيانات المختلفة في النوع مع بعضها البعض بحيث يمكن التعامل معها كوحدة واحدة .

النوع : أي أن يكون بعضها من نوع int وبعضها char وبعضها float وبعضها ....الخ.

فمثلاً :
لكتابة برنامج لتسجيل بيانات موظفين في الشركة نحتاج إلى تخزين :

١- اسم الموظف وهو من نوع مصفوفة حرفية char name[40]
٢- عنوانه وهو من نوع مصفوفة حرفية char address[40]
٣- عمره متحول من نوع عدد صحيح int age
٤- راتبه متحول من نوع عدد حقيقي float salary

وكما نلاحظ فإن جميع هذه البيانات يجب التعامل معها كوحدة واحدة لأنها لموظف واحد ولذلك فإننا بحاجة إلى سجل خاص لهذا الموظف

كيفية الإعلام عن السجل :
للتصريح عن سجل نستخدم الكلمة المحجوزة struct وهي اختصار لكلمة structure ومعناها تركيب

ونضع جميع مكونات هذا التركيب ضمن قوسين ونختم التصريح بفاصلة منقوطة بعد القوس الثاني
ثم يتم التصريح عن متحول خاص لهذا التركيب (السجل) ضمن القائمة الرئيسية للبرنامج .
ويتم التصريح عن السجل بأحد الطرق التالية

```
#include <iostream.h>
struct  employee        اسم السجل
{
---------;
---------;
---------;
---------;
};                      لا ننسى الفاصلة المنقوطة هنا
Void main ( )
{
struct  employee  emp;      هنا نصرح عن متحول خاص بالسجل
```

كما نلاحظ
فإننا كتبنا السجل وكتبنا بداخله جميع المتحولات الموجودة بداخله
ومن ثم بدأنا بكتابة البرنامج بالعبارة المعتادة void main ( ) ومن ثم صرحنا
عن متغير خاص بهذا السجل – طبعا نختار اسم المتحول كيفيا – ومن ثم نتابع كتابة البرنامج

كيف نكتب المتحولات داخل السجل (كيفية كتابة هذا السجل)

```
#include <iostream.h>
struct  employee
{
char name[40];          مصفوفة لكتابة اسم الموظف
char address[40];   مصفوفة لكتابة عنوانه
int age;            متحول لكتابة عمره
float salary;       متحول لكتابة راتبه
};
```

ونصرح طبعاً عن متحول خاص بهذا السجل كالتالي:

```
Void main ( )
{
struct  employee emp;
```

كيفية إدخال وإخراج المعلومات داخل هذا السجل

عندما نريد مثلاً إدخال عمر الموظف فإننا نكتب

Cin >> emp . age ;

وعندما نريد إدخال اسم الموظف نكتب

Cin >> emp . name;

و لا نكتب الأقواس عندما نريد إدخال الاسم كاملاً كما تعلمنا في المصفوفات

أما عندما نريد إخراج أي قيمة فإننا نستبدل cin بـ cout كما هو معروف

ويمكننا أن نعلم عن التصريح بطريقة أخرى

```
Void main ( )
{
Struct employee
{
-----;
-----;
-----;
-----;
};
ونكمل البرنامج
```

مثال عملي:

اكتب برنامج يقوم بإدخال معلومات عن

١- اسم الموظف
٢- عنوانه
٣- عمره
٤- راتبه

```
#include<iostream.h>
struct employee            طبعا لا نضع هنا فاصلة منقوطة
{
char name[40];
char address[40];
int age;
float salary;
};
void main()
{

struct employee emp;
cout<<"enter name"<<endl;
```

```
cin>>emp.name;
cout<<"enter address"<<endl;
cin>>emp.address;
cout<<"enter age"<<endl;
cin>>emp.age;
cout<<"enter salary"<<endl;
cin>>emp.salary;
}
```

ولكن إن كان لدينا أكثر من موظف مثلاً ثلاثة عشر موظف فإننا بحاجة إلى مصفوفة سجلات

كيفية التصريح عن مصفوفة سجلات

نفس الطريقة السابقة ولكن نجعل المتحول عبارة عن مصفوفة ونضيف سطر برمجي وهو حلقة for كما كنا نكتبها في المصفوفات .

مثال عملي:
لنعد كتابة البرنامج السابق ولكن لثلاثة عشر موظفاً ندخل المعلومات عنهم ومن ثم نخرجها

```
#include<iostream.h>
struct employee
{
char name[40];
char address[40];
int age;
float salary;
};
void main()
{
struct employee emp[13];
for(int i=0;i<13;i++)
    {
          cout<<"enter name"<<endl;
          cin>>emp[i].name;
```

```
            cout<<"enter address"<<endl;
            cin>>emp[i].address;
            cout<<"enter age"<<endl;
            cin>>emp[i].age;
            cout<<"enter salary"<<endl;
            cin>>emp[i].salary;

    }
for(i=0;i<13;i++)

cout<<emp[i].name<<emp[i].address<<emp[i].age<<emp[i].salary<<endl;
 }
```

سؤال:
إن أردنا توسيع هذه المسألة مثلاً أن تكون مجموعة من الموظفين تابعة لقسم ما أو لشركة ما فماذا نفعل؟
نحن هنا بحاجة إلى تعريف سجل داخل سجل.

تعريف سجل داخل سجل آخر

من أجل تعريف سجل داخل سجل من أجل ارتباط السجل الثانوي بالسجل الرئيسي مثل أن نكتب برنامج يدخل اسم الموظف وعنوانه و.....و..... ويتم إسناد هذا السجل إلى سجل آخر وهو القسم أو المؤسسة التي يوجد بها هذا الموظف
فيكون هنا سجل المؤسسة هو السجل الرئيسي أما سجل الموظف فهو السجل الثانوي
وليس من الضروري - في كتابة الكود - كتابة أي سجل قبل الآخر

مثال على ذلك :

```
Struct employee      سجل الموظف ثانوي
{
Char name[20];
Char address[40];
Int age;
Float salary;
```

```
};
Struct dept            سجل القسم رئيسي
{
Int deptno;            نعرف متحول رقم القسم
Int projct;            نعرف متحول المشروع
Struct employee emp;   نعرف هنا متحول السجل الثانوي داخل الرئيسي
};
```

أما عن كيفية الإدخال و الإخراج فإننا نكتب
فإذا أردنا إدخال مثلاً عمر الموظف فإننا نكتب اسم السجل الرئيسي ثم المتحول للسجل الثانوي ثم مكان الإدخال أما في حال إضافة معلومة ضمن السجل الرئيسي فيكفي كتابة اسم السجل الرئيسي ثم كتابة اسم مكان الادخال

مثال :

Dept.emp.age;
Dept.deptno

ملاحظة:
من أجل إدخال سلسلة حرفية – فقط - يمكن استخدام التعليمة gets بدل التعليمة cin>> في حال استخدام المكتبة التابعة لهذه التعليمة وهي <stdio.h> ويكتب الكود كالتالي:

```
gets (emp[i] .name);
```

Function

هو عبارة عن برنامج فرعي مهمته تنفيذ مهمة معينة حين يتم استدعاؤه

مما يتألف البرنامج الفرعي

يتألف من ثلاثة أقسام

١- التصريح عن البرنامج (وسيأتي شرحه بعد قليل)وينتهي بفاصلة منقوطة
ولكن إن كان التصريح خارج (قبل) التعليمة

Void main ( )

فإن البرنامج الفرعي يكون مشاع (لكل البرامج المتاحة في البرنامج)
أما إن كتب داخل جسم

Void main ( )

فإن البرنامج الفرعي يكون حكراً على هذه التعليمة .

٢- كود طلب البرنامج وينتهي بفاصلة منقوطة (يشرح من خلال الأمثلة)

٣- جسم البرنامج وهو يشابه طريقة كتابة البرنامج الرئيسي وطبعاً بدون فاصلة منقوطة ويكتب جسم التابع بعد نهاية كتابة جسم البرنامج الرئيسي

كيفية التصريح عن هذا التابع:

وهو بأن نكتب اسم خرج التابع ثم اسم التابع ومن ثم نكتب نوع البارامترات أي نوع الخرج

خرج التابع	اسم التابع	البارامترات
Void	positive	(int) ;

ويمكن أن يكون خرج التابع

فارغ	void
عدد صحصح	int
بدون قيمة	void
عدد حقيقي	float

....

....

.....الخ .

مثال:(بدون عملية إرجاع أي قيمة) أي بدون حاجة إلى خرج

اكتب برنامج يقوم بمعرفة العدد - المُدخَل من قبل المستخدم- موجب أم سالب

```
#include<iostream.h>
void positive (int);        لا ننسى كتابة الفاصلة المنقوطة
void main()
{
int x;
do
{
cin>>x;
positive (x);  كود طلب البرنامج   استدعاء البرنامج الفرعي
}
while(x !=0);
}
void positive (int a)    البرنامج الفرعي
{
if(a<0)
cout<<a<<"is a negative number";
else
cout<<a<<"is the positive number ";
}
```

شرح البرنامج:
تم التصريح عن البرنامج بالكود التالي
void positive (int);
ومعناه بأن البرنامج غير مطالب بإرجاع (خرج البرنامج الفرعي) إلى البرنامج الأساسي من خلال التعليمة void
ثم من خلال حلقة do while طالبنا البرنامج بالعمل طالما أن المتحول x لا يساوي الصفر
وتم استدعاء البرنامج الفرعي من خلال الكود التالي
positive (x);
فبواسطة هذه التعليمة فإن البرنامج سيقفز مباشرة إلى البرنامج الفرعي و المتمثل بـالتالي

```
void positive (int a)
{
if(a<0)
cout<<a<<"is a negative number";
else
cout<<a<<"is the positive number ";
}
```

لقد عرفنا هنا المتحول a بمتحول شكلي و تسند قيمة المتحول الفعلي x المدخلة عليه ومن ثم نفذنا عليه هذا البرنامج الفرعي
وبعد الانتهاء من البرنامج الفرعي يعود البرنامج لمتابعة البرنامج الرئيسي و المتمثل هنا إعادة إدخال القيمة x

ملاحظة هامة:
يوجد طريقتين للتصريح عن التابع
١- أن نقوم بالتصريح عن التابع ومن ثم نقوم بكتابة التابع بشكل كامل بعد نهاية void mainبالكامل أي بعد نهاية القوس الثاني.
٢- أن نقوم مباشرة بالتصريح عن التابع وكتابة جسم التابع مباشرة ولا نكتب التصريح

عند تنفيذ مثل هذا البرنامج فإن البرنامج سينفذ void main أول شيء

مثال: مع الحاجة إلى إرجاع
اكتب برنامج يكتشف أكبر عدد من ثلاثة أعداد مدخلة عليه.

```
#include<iostream.h>
int great (int,int,int);     نحن هنا بحاج للتعامل مع ثلاثة بارامترات
void main()
{
```

```
int a,b,c,d;
cin>>a>>b>>c;
d=great(a,b,c);     كود طلب البرنامج الفرعي
cout<<d;
}
int great(int x,int y,int z)    البرنامج الفرعي
{
int max;
max=x;
if(y>max)  max=y;
if(z>max)  max=z;
return max;          تعيد هذه التعليمة البرنامج إلى آخر قيمة للمتحول المكتوب
}                    بعدها
```

إذا لم تكن لدينا قيمة معادة فإننا نحذف السطر return max
ونكتب بدلا من int العبارة void في السطرين

```
 int great (int,int,int);
int great(int x,int y,int z)
```

ملاحظة هامة جداً

* نلاحظ أن التابع في هذا البرنامج يقوم بإعادة قيمة واحدة إلى البرنامج المستدعي فللاستفادة من هذه القيمة المرجعة يجب استخدام متحول آخر تسند إليه قيمة خرج هذا التابع

d=great(a,b,c);

* عندما يكون التابع يعيد قيمة فإن هذه القيمة سوف تعاد ضمن الأمر retun حيث أن وظيفة هذه التعليمة هي تمرير القيمة المكتوبة بعدها إلى البرنامج المستدعي

مثال :

اكتب برنامج يقوم برفع عدد الى قوة بدون استخدام الأمر pow(x,n) الممكن ضمن المكتبة

<math.h>

مع العلم أن abs(b) عبارة عن تابع يعطي القيمة المطلقة للقيمة (b)

الحل:

```
#include<iostream.h>
#include<math.h >
float power(float,int);
void main()
```

```
{
float x; int n;
cin>>x>>n;
cout<<power (x,n);            اطلب البرنامج الفرعي وطبع القيمة المعادة
}
float power (float a,int b)   البرنامج الفرعي
{
float c; int i;
c=1;
for(i=abs(b); i>1;i--)
c*=a;
if(b<0) c=1/c;
return c;
}
```

هام جداً

عند وجود قيمة معادة يجب استخدام التعليمة return

تعاد في هذا البرنامج قيمة المتحول c إلى البرنامج الرئيسي

اكتب برنامج من أجل حساب و طباعة قيمة العلاقة التالية

Sum = 0! + 1! + 2! + 3! + 4! + 5!

الحل:

```
#include<iostream.h>
int fact (int);
void main ()
{
int sum=0,i;
for(i=0;i<=5;i++)
sum+=fact(i);            طلب خرج البرنامج الفرعي خمس مرات وجمع القيم المرجعة
cout<<"sum = "<<sum;
}
int fact (int n)
{
int i,s;
s=1;
for(i=n;i>=1;i--)
s*=i;
```

```
return s;
}
```

اكتب برنامج من أجل حساب وطباعة قيمة التعبير التالي

Sum = 1 + x^1 + x^2 + x^3 + x^4 + x^5

مع العلم أن (x^n) تعني أن الرقم اكس مرفوع إلى القوة n

باستخدام الأمر ;pow(x,i)

```
#include<iostream.h>
#include<math.h>
void main()
{
int sum=0,x,i;
 cin>>x;
for(i=5;i>=1;i--)
sum+=pow(x,i);
cout<<sum;
}
```

FILES الملفات

الملفات هي عبارة عن وسائط لتخزين البيانات بحيث يتم الكتابة عليها و تحفظ هذه البيانات بداخلها

سنقوم بهذا الدرس بشرح كيفية التعامل مع الملفات من حيث القراءة و الكتابة على أن نستعين ببعض مكتبات لغة C

شرح بعض تعليمات لغة C

أولاً يجب أن نعلم أن المكتبة الداعمة لهذه الأوامر هي <stdio.h >

تعليمة فتح ملف fopen ( )
تعليمة إغلاق ملف fclose ( )
تعليمة طباعة ملف fprintf ( )

الشرح

لفتح ملف يجب علينا تحديد مسار هذا الملف و امتداده

ويجب علينا أيضاً تحديد مكان في الذاكرة لحجزه لهذا الملف.بالتعليمة FILE *f
ويجب علينا الانتباه بأن الكلمة FILE يجب أن تكتب بأحرف كبيرة حصراً
وعند فتح هذا الملف يجب علينا إخبار البرنامج بأننا هل نريد القراءة من هذا الملف أم الكتابة أم الاثنتين معاً .

فلكتابة نستخدم f = fopen("c:\\student.txt","w");

أما للقراءة فنستخدم "r"

وفي حال القراءة و الكتابة نستخدم "r+" أو "w+"

مثال :

```
#include <stdio.h >
void main ( )
{
FILE * f
f=fopen("c:\\stdudent.txt","w");
fprintf(f," welcome to C++");
fclose(f);
}
```

ولطباعة welcome to C++ في هذا الملف الذي تم فتحه أو إنشاءه نستخدم الكود التالي

```
fprintf(f," welcome to C++");
```

وإن التعليمة fprint تكافئ التعليمة cout في المكتبة iostream
ومن أجل إغلاق هذا الملف استخدمنا هذا الكود

```
fclose(f);
```

وإن هذه التعليمة تقوم بإغلاق هذا الملف وتنهي التعامل معه وعدم كتابة هذه التعليمة يمكن أن يؤدي في بعض الأحيان إلى تلف أو ضياع البيانات .

ملاحظة :

* عند الطلب من البرنامج أمر فتح ملف فإن البرنامج إن لم يجد هذا الملف فإن البرنامج سينشئ هذا الملف.
* في حال عدم كتابة مسار الملف المنشأ من قبل البرنامج فإن البرنامج سينشئ هذا الملف في المسار الافتراضي للبرنامج (المسار الموجود به البرنامج).

مثال :

اكتب برنامج يقوم بإنشاء ملف نصي من أجل تخزين علامات طالب ضمن هذا الملف

```
#include<iostream.h>
#include<stdio.h>
#include<stdlib.h>
void main()
```

```
{
FILE *out;    حجز مكان للملف المنشأ
int score;
if((out=fopen("test.txt","w"))==NULL)
{
cout<<"can not open file\n";
exit(-1);           هذه التعليمة تقوم بإنهاء البرنامج
}
cout<<"enter a test score (0 terminate input)";
cin>>score;
while(score !=0)
{
fprintf(out,"%d\n",score);
cout <<"enter anather score";
cin>>score;
}
fclose(out);
}
```

شرح البرنامج

بداية تم استخدام مكتبة جديد ة هي stdlip لاستخدام بعض الأوامر منها

سنكتب الأكواد بالترتيب مع الشرح

FILE *out;

تحديد مكان في الذاكرة للتعامل مع الملف وأطلقنا عليه اسم out

وكما ذكرنا سابقا كلمة FILE تكتب بأحرف كبيرة حصراً

int score;

تعريف متحول على أنه عدد صحيح

if((out=fopen("test.txt","w"))==NULL)

معنى NULL (وتكتب بأحرف كبيرة حصراً) فشل ، إحباط و هي كلمة محجوزة

فيكون الكود :

إذا لم يتم فتح هذا الملف لسبب من الأسباب (مثل : القرص محمي ضد الكتابة) نفذ السطر التالي في البرنامج وإلا فتخطاه

exit(-1);

قم بإنهاء البرنامج و هي من المكتبة stdlib

إن حلقة if متضمنة الأمرين معا :

إعطاء رسالة الخطأ و الخروج من البرنامج

cin>>score;

أمر إدخال قيمة المتحول وهو إدخال علامة الطالب

while(score !=0)

طالما أن العلامة لا تساوي الصفر نفذ الأوامر التالية
وقد كتبت من أجل تضمين البرنامج أمر ليتسنى لنا الخروج من البرنامج

fprintf(out,"%d\n",score);

تفيد هذه التعليمة أن ما يكتب في الملف هو من نوع int

fclose(out);

وهو أمر لإغلاق الملف.

كيفية التعامل بين الملفات و السجلات

وتكون بنفس طريقة طباعة الملف مع مراعاة كيفية كتابة الأمر fprintf و يكون كتالي
fprintf(اسم الملف المحجوز بالذاكرة , " % s" , اسم متحول السجل . اسم السجل) ;
فمثلاً : ليكن لدينا السجل التالي :

```
struct employee
{
int empno;
char name[20];
float salary;
};
```

فإذا أردنا طباعة مصفوفة الاسم name[20] داخل الملف فإننا نكتب الكود

```
fprintf (myfile %s ", emp[i].name);
```

أما إن أردنا طباعة هذا السجل بالكامل فإن الكود يكون (على سطر واحد يكتب هذا الكود)

```
fprintf
(myfile,"%d\t%s\t%f\n",emp[i].empno,emp[i].name,emp[i].salary);
```

ملاحظة

%d تكافئ في لغة ++C	int
%s	string
%f	float
%c	char

كما نلاحظ بأننا عند التعامل مع الملفات نحن بحاجة للتعامل مع المكتبة < stdio.h>
مثال :
اكتب برنامج يقوم بقراءة بيانات الموظفين ومن ثم يقوم بتخزينها في الملف out.txt علماً أن البيانات تتضمن رقم الموظف واسمه وكذلك راتبه
*سنشرح الأكواد داخل البرنامج
*ملاحظة :
بعض الأكواد طويلة ولم تتسع معنا بسطر واحد ولم نرغب بتصغير الخط ولذلك يرجى الانتباه مع العلم بأننا نبهنا إلى ذلك في مكان كتابة الكود

```
#include<iostream.h>
#include<stdio.h>
#include<stdlib.h>
void main()
{
FILE *myfile;          حجز مكان في الذاكرة بهذا الاسم
struct employee        كتابة السجل
{
int empno;             تعريف رقم الموظف داخل السجل
char name[20];         تعريف مصفوفة اسم الموظف داخل السجل
float salary;          تعريف راتب الموظف داخل السجل
};
employee emp[100];     تعريف مصفوفة متحول للسجل
int i,n;
if((myfile=fopen("out.txt","w"))==NULL)
{
cout<<"can not open file";
exit(-1);
cout<<"enter the number of employee<=100";
cin>>n;
for(i=0;i<n;i++)       تحديد عدد السجلات المراد إدخالها
{
```

```
cout<<"enter the number of employee"<<i<<" : ";
cin>>emp[i].empno;
cout<<"enter the name of employee"<<i<<" : ";
cin>>emp[i].name;
cout<<"enter the salary of employee"<<i<<" : ";
cin>>emp[i].salary;
fprintf
(myfile,"%d\t%s\t%f\n",emp[i].empno,emp[i].name,emp[i].salary);
```

يكتب هذا الكود في سطر واحد فيرجى الانتباه
وهو أمر طباعة السجلات داخل الملف بواسطة أمر الطباعة

```
}
fclose(myfile);      أمر إغلاق الملف
}
```

القراءة والكتابة على ملف ببرنامج واحد

مثال

اكتب برنامج يقوم بقراءة بيانات الموظفين وذلك من الملف المنشأ سابقاً out.txt ومن ثم يقوم البرنامج بترتيبها حسب الاسم ثم يقوم بإعادة تخزينها بنفس الملف

```
#include<iostream.h>
#include<stdio.h>
#include<stdlib.h>
#include<string.h>      مكتبة جديدة لاستخدام أمر المقارنة و أمر الاستفسار
void main()
{
FILE *myfile;
struct employee
{
int empno;
char name[20];
float salary;
```

```
};
employee emp[100],temp;
int i=0,j,n=0;
if((myfile=fopen("out.txt","r+"))==NULL)
{
cout<<"can not open file";
exit(-1);
}
```

شرح سابقاً

يجب الانتباه بأن الكود التالي يكتب على سطر واحد

```
fscanf(myfile,"%d%s%f",&emp[i].empno,&emp[i].name,&emp[i].salary);
```

أمر فتح الملف ويجب الانتباه لعلامة الربط & بين المتحولات

```
while(!feof (myfile))
```

feof تعني finch in of file ومعناها هل انتهى الملف ويكون الشرح طالما لم يتم الوصول لنهاية الملف تابع

```
{
i++;      زيادة واحد للوصول للسطر التالي لنقل المعلومات منه
n++;      عداد لمعرفة عدد الاسطر
```

يكتب الكود على سطر واحد

```
fscanf(myfile,"%d%s%f",&emp[i].empno,&emp[i].name,&emp[i].salary);
```

فتح سطر كما تم ذكره ولكن هنا يكون هذا السطر هو السطر التالي كل مرة ضمن الحلقة

```
}
for(i=0;i<n;i++)
for(j=i+1;j<n;j++)
if(strcmp (emp[i].name,emp[j].name)>0)
```

strcmp تعليمة مقارنة وتكون النتيجة أكبر من الصفر في حال كانت القيمة الأولى أكبر من القيمة الثانية
وتكون مساوية للصفر إذا تساوت القيمتين
وأصغر من الصفر إذا كانت الثانية أكبر من الأولى

```
{
temp=emp[i];          كود ذكي لتبديل مكان السجلات
emp[i]=emp[j];
emp[j]=temp;
}
for(i=0;i<n;i++)      حلقة من أجل الكتابة الى الملف
```

```
fprintf(myfile,"%d\t%s\t%f\t",emp[i].empno,emp[i].name,emp[i].salary);
fclose(myfile);
}
```

شرح الكود الذي اسميناه ذكي
نضع السجل emp [i] داخل السجل temp ليتسنى لنا
نقل السجل emp [j] الى داخل السجل emp [i] ومن ثم
ننقل السجل temp الى داخل السجل emp [j]
وبهذا نكون قد أبدلنا مكانا السجل emp [i] بالسجل emp [j]

تم بحمد الله الانتهاء من كتابة هذه النوتة وإن شاء الله هي خالية من الأخطاء
وجل من لا يسهى
ونرجو منكم المتابعة بها وإعادة نشرها
ونرجو الدعاء لنا في ظهر الغيب ولله المبتغى من وراء القصد

أخوكم في الله

محمد هنداوي ٣٠ / ٨ / ٢٠٠٦
mh2n@scs-net.org